



PATENT ABSTRACTS OF JAPAN

(11) Publication number: **61127044 A**(43) Date of publication of application: **14.06.86**

(51) Int. Cl.

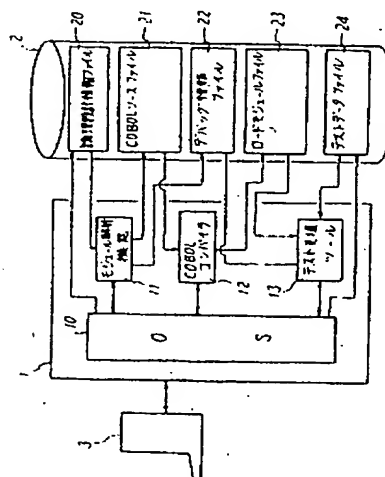
G06F 11/28**G06F 9/44**(21) Application number: **59250174**(71) Applicant: **FUJITSU LTD**(22) Date of filing: **27.11.84**(72) Inventor: **YANAGISAWA MINORU**(54) **TEST SYSTEM FOR SINGLE UNIT PROGRAM**

COPYRIGHT: (C)1986,JPO&Japio

(57) Abstract:

PURPOSE: To improve the developing efficiency of a program by providing a memory to a test back-up tool which replaces modules to start a module to be tested after the test data is set or setting and delivering the module during its test.

CONSTITUTION: A test back-up tool 13 is provided to a processor 1 connected to a console 3 in addition to a module analysis function 11 and COBOL compiler 12. An operating system 10 controls these function 11, compiler 12 and tool 13 respectively. A test data file 24 is provided to a file memory 2 and then connected to the tool 13. The test data on the module to be tested is stored in the file 24 and the tool 13 is started. A parameter delivery area is secured in an internal memory of the processor 1 if a high-order module exists. Then the data is set to start a test. While said delivery area is secured in the internal memory when no low-order load module exists. Then the data is set through the file 24 and used as the output data of low-order module.



⑩ 日本国特許庁(JP)

⑪ 特許出願公開

⑫ 公開特許公報(A)

昭61-127044

⑬ Int. Cl.⁴

識別記号

庁内整理番号

⑭ 公開 昭和61年(1986)6月14日

G 06 F 11/28
9/44

7343-5B
Z-8120-5B

審査請求 未請求 発明の数 1 (全6頁)

⑮ 発明の名称 プログラムの単体テスト方式

⑯ 特 願 昭59-250174

⑰ 出 願 昭59(1984)11月27日

⑱ 発 明 者 柳 沢 実 川崎市中原区上小田中1015番地 富士通株式会社内

⑲ 出 願 人 富士通株式会社 川崎市中原区上小田中1015番地

⑳ 代 理 人 弁理士 山谷 皓 榮

明 細 書

1. 発明の名称 プログラムの単体テスト方式

2. 特許請求の範囲

複数のモジュールが階層的に構成されるプログラムのモジュール単体をプロセッサが実行してテストを行うプログラムの単体テスト方式において、該被テストモジュールを解析して該被テストモジュールに対する上位又は下位のモジュールの代替機能を行うテスト支援ツールを設け、該テスト支援ツールは該被テストモジュールが該上位モジュールから呼出される場合には、該被テストモジュールに渡すパラメータの領域をメモリに確保してテストデータをセット後該被テストモジュールを起動せしめ、該被テストモジュールが該下位モジュールを呼出す場合には、該被テストモジュールのテスト実行中に該下位モジュールが呼出されたことに応じて被テストモジュールに渡すテストデータを該メモリにセットして該被テストモジュール

に渡すことを特徴とするプログラムの単体テスト方式。

3. 発明の詳細な説明

[産業上の利用分野]

本発明は、複数のモジュールで構成されるプログラムの各モジュールプログラムを単体でテストするプログラムの単体テスト方式に関し、特に関連する他のモジュールが作成されていなくてもテストできるプログラムの単体テスト方式に関する。

一般に大きなシステムにおいては、プログラムが1本のモジュール(コンパイル単位)で構成されているものは少なく複数のモジュールが集まって一つの処理を行なうように階層的に構成されている。

例えば、図5図に示すようにモジュールAのサブルーチンとしてモジュールBが、モジュールBのサブルーチンとしてモジュールC、Dが構成されて1つのプログラムを構成するようにしている。このような階層的なプログラムを開発するには、テストの効率を考慮してモジュールの作成順序等が決定され、一般に図6図に示す如くシステム設

計された後、各モジュールA、B、C、Dが作成される。

この開発の手順としては、上位のモジュールAから作成するトップダウン方式と、下位のモジュールC又はDから作成するボトムアップ方式とがある。

〔従来の技術〕

このような各モジュールが階層的に構成されたプログラムをテストするには、全てのモジュールが完成してからプログラムをテストすることが望ましいが、各モジュールが順次作成されていくような場合には、各モジュールの作成毎にそのモジュールを単体でテストするほうが開発手順上効率がよい。

しかしながら、モジュールを単体でテストするにしても、階層的なプログラムの構成では、関連する他のモジュールが存在しないとテストできない。例えばモジュールAをテストするときは、モジュールBが必要であり、モジュールBをテストするときは、モジュールA、C、Dが必要となる。

的に構成されるプログラムのモジュール単体をプロセッサが実行してテストを行うプログラムの単体テスト方式において、該被テストモジュールを解析して該被テストモジュールに対する上位又は下位のモジュールの代替機能を行うテスト支援ツールを設け、該テスト支援ツールは該被テストモジュールが該上位モジュールから呼出される場合には、該被テストモジュールに渡すパラメータの領域をメモリに確保してテストデータをセット後該被テストモジュールを起動せしめ、該被テストモジュールが該下位モジュールを呼出す場合には、該被テストモジュールのテスト実行中に該下位モジュールが呼出されたことに応じて被テストモジュールに渡すテストデータを該メモリにセットして該被テストモジュールに渡すことを特徴としている。

〔作用〕

本発明では、テスト支援ツール（プログラム）が設けられるが、この支援ツールがドライバ又はスタブ機能を自動的に実行するようにしている。

このため、従来は、テストするモジュールに関連する仮モジュールをテスト担当者が作成し、係るモジュールの単体テストを行っていた。

例えば、モジュールBをテストするため上位のモジュールAの仮モジュール（ドライバという）及び下位のモジュールC、Dの仮モジュール（スタブという）を作成していた。

〔発明が解決しようとする問題点〕

このようなモジュールのテストで必要となるドライバやスタブは仮モジュールなので小さく簡単に作成できるが、通常のプログラム作成と同様にコーディングやこれのテストの工数を要し、単体テストのためのドライバやスタブの開発をモジュール個々に要するという問題があった。

〔問題点を解決するための手段〕

本発明は、テストされるモジュールの必成とするドライバ又はスタブを自動的に発生させ、仮モジュールの作成を必要としないで単体テストできるプログラムのテスト方式を提供するにある。

このため、本発明は、複数のモジュールが階層的

に構成されるプログラムのモジュール単体をプロセッサが実行してテストを行うプログラムの単体テスト方式において、該被テストモジュールを解析して該被テストモジュールに対する上位又は下位のモジュールの代替機能を行うテスト支援ツールを設け、該テスト支援ツールは該被テストモジュールが該上位モジュールから呼出される場合には、該被テストモジュールに渡すパラメータの領域をメモリに確保してテストデータをセット後該被テストモジュールを起動せしめ、該被テストモジュールが該下位モジュールを呼出す場合には、該被テストモジュールのテスト実行中に該下位モジュールが呼出されたことに応じて被テストモジュールに渡すテストデータを該メモリにセットして該被テストモジュールに渡すことを特徴としている。

〔実施例〕

以下、本発明を実施例により詳細に説明する。

図1図は本発明の一実施例ブロック図であり、図中、1はコンピュータ（プロセッサ）であり、プログラムを実行して所望の処理を行なうもの、

10はそのOS(オペレーティングシステム)部であり、コンピュータ1全体の制御を行なうもの、11はモジュール解析部であり、後述する如く論理設計言語で記述された論理設計情報を解析してCOBOLのソースプログラムを作成し、且つインターフェイス(デバッグ)情報を抽出するもの、12はCOBOLコンパイラであり、ソースプログラムをコンピュータの実行できる形式のロードモジュールに変換するもの、13はテスト支援ツール部であり、後述する如くインターフェイス情報からドライバ/スタブ機能を行なうものである。尚、OS部10、モジュール解析部11、COBOLコンパイラ12及びテスト支援ツール部13はコンピュータ1の実行する機能をプロダク化したものである。2はファイルメモリであり、各種ファイルを格納しておくもの、20は論理設計情報ファイルであり、プログラマーが作成した後述する論理設計情報を格納しておくもの、21はCOBOLソースファイルであり、モジュール解析部11によつて作成されたCOBOLソースプロ

グラムを格納しておくもの、22はデバッグ情報ファイルであり、モジュール解析部11で抽出されたインターフェイス情報を格納しておくもの、23はロードモジュールファイルであり、COBOLコンパイラ12が作成したロードモジュールを格納しておくもの、24はテストデータファイルであり、オペレータが入力したテストデータを格納しておくもの、3はコンソールであり、ディスプレイとキーボードから成り、オペレータが操作して論理設計情報やテストデータを入力するものである。

なフォーマット定義書(オ3図(A))、各モジュールのインターフェイス定義書(オ3図(B))、各モジュールのモジュール定義書(オ3図(C))等を作成することにより行なう。この共通フォーマット定義書は各モジュールで共通な項目(データ)の形式、指数等の定義を行なうものであり、インターフェイス定義書は各モジュールの入出力項目、モジュール名等を定義するためのものであり、基本設計段階で決定され、コンソール3よりコンピュータ1を介しファイルメモリ2の論理設計情報ファイルに格納されている。

一方、モジュール定義書は各モジュール毎に作成され、当該モジュールの処理や論理、その条件が手順に従つて羅列されたものであり、プログラマーは各モジュールに必要な処理等を順次記述してプログラミングし、これが当該モジュールの処理内容を示す。オペレータはこのモジュール定義書を完成する毎にコンソール3からコンピュータ1を介し論理設計情報ファイル20に格納せしめる。

次に、オ1図実施例構成の動作についてオ2図の処理フロー図、オ3図の論理設計情報説明図及びオ4図のテスト支援ツールの動作説明図を用いて説明する。

以下、周知の高級言語である論理設計言語によりプログラミングされたものについてその動作を説明する。

① 論理設計言語によるプログラミングは、オ3図に示す如く各モジュールA、B、C、Dで共通

例えば、モジュールBのモジュール定義書が完成し、論理設計情報ファイル20に格納され、他のモジュールA、C、Dは完成していないとする。

② オペレータはコンソールよりコンピュータ1にOS部10のジョブ制御機能を働かせ、モジュールBのCOBOLソースプログラムを作成するように指令する。これによつてコンパイラの一つであるモジュール解析部11は論理設計情報ファイル20の論理設計情報を読出し、COBOLで記述されたソースプログラムを作成し、COBOLソースファイル21に格納する。このCOBOLで記述されたソースプログラムは、例えばオ4図の被テストモジュールとして示すものであり、論理設計言語を用いずプログラマーがCOBOLでプログラミングしたものに相当する。

③ これとともにモジュール解析部11は論理設計情報からインターフェイス情報を抽出する。例えば、オ3図の例では、モジュールBはモジュールAに呼び出され、パラメータの数はG1、G2の2つでその領域はA1、A2であることがモジ

ジュールBのインターフェイス定義書より抽出され、そのパラメータG1, G2の長さは共通フォーマット定義書より抽出される。又、モジュールBのモジュール定義書よりモジュールO(日数=%日数計算)、モジュールDを呼び出すことが抽出され、その使用領域はモジュールO, Dのインターフェイス定義書より抽出される。

これをCOBOLソースプログラムの記述から見ると、オ4図の被テストモジュールの記述の如く、モジュールAのA1, A2を用い、モジュールCのB1, モジュールDのB2を用いることになる。

モジュール解析部11はモジュールBが呼び出されるときにインターフェイス(モジュール名、パラメータの数、パラメータの領域名、その長さ等)及びモジュールBが呼び出す時のインターフェイスを論理設計情報から抽出し、デバッグ情報ファイル22に格納する。

このようにしてモジュールBに対するCOBOLソースプログラムが作成され、且つモジュール

トモジュールBに渡すパラメータの受け渡し領域をコンピュータ1の図示しない内部メモリに確保する。前述の例では、オ4図に示す如く内部メモリにA1及びA2の入出力領域を確保し、そのアドレスをセフトしておく。

次に、コンソール3よりオペレータにテストデータの指示を表示し、オペレータはテストデータファイル24のどのテストデータをセフトするかをコンソール3より指示する。これによつて内部メモリのA1の領域にテストデータファイル24の必要なテストデータ(上位モジュールからの入力データ)がセフトされることになる。

これによつてモジュールBはパラメータA1を用いて処理を実行できる。尚、A2はモジュールBの実行結果の出力のためのものである。

④ 次に、テスト支援ツール部13はOS部10に対し被テストモジュールの起動を指示する。

OS部10はロードモジュールファイル23のモジュールBのロードモジュールを実行し、その中でモジュールAのパラメータA1を用いて行な

間のインターフェイス情報が抽出される。

④ 次に、モジュールBの単体テストを行なう。

このため、オペレータはコンソール3よりコンピュータ1を介しモジュールBのテストのためのテストデータをテストデータファイル24に格納する。そして、オペレータはテスト実行指令をコンピュータ1に入力する。コンピュータ1では、OS部10がCOBOLコンパイラ12を動作せしめ、COBOLソースファイル21のモジュールBのCOBOLソースプログラムをコンピュータ1の実行できるロードモジュールに翻訳してロードモジュールファイル23に格納する。次にOS部10はテスト支援ツール部13を起動し、デバッグ情報ファイル22のインターフェイス情報を解析せしめる。

⑤ テスト支援ツール部13は、オ2図の処理フロー図の如く前述のインターフェイス情報を解析し、上位モジュールが存在するかを検出する。上位モジュールが存在するとドライバ機能、即ち上位モジュール代替機能を実行する。先づ、被テス

う。

⑥ このようにしてモジュールBのテストを実行していくうちに、モジュールO又はDの"call"(呼び出し)命令があり、OS部10がロードモジュールファイル23を探索した結果モジュールO, Dのロードモジュールが見付からなかつたとすると、エラーをテスト支援ツール部13に通知する。

⑦ これによつてテスト支援ツール部13は、被テストモジュールが下位モジュールを呼ぶために用意した内部メモリのパラメータ領域(B1やB2)のアドレスを得、コンソール3よりオペレータにテストデータの指示を表示する。オペレータはテストデータファイル24のどのテストデータをセフトするかをコンソール3より指示する。これによつて内部メモリのパラメータ領域にテストデータファイル24の必要なテストデータ(下位モジュールの出力データ)がセフトされることになる。

従つて被テストモジュールBはこのテストデー

タを下位モジュールの出力データとして受け取り、更にテストを続行していく。

このようにして、テスト支援ツール部13が、インターフェイス情報から上位又は下位モジュールの代替機能を自動的に実行し、被テストモジュールのテストを行なわしめる。

上述の実施例では、論理設計言語でプログラムした例について説明したが、COBOLによってプログラムしてソースプログラムを作成してもよく、この場合COBOLソースプログラムからインターフェイス情報が抽出される。又、テストデータをパラメータ領域にセットするのにコンソールよりオペレータが指示しているが、これを自動的に行なってもよい。更にソースプログラムの言語もCOBOLに限らず、他の言語を用いてもよい。

以上本発明を一実施例により説明したが、本発明は本発明の主旨に従い種々の変形が可能であり、本発明からこれらを排除するものではない。

〔発明の効果〕

の開発効率を高めることができる。又、被テストモジュールに対するドライバ、スタブをプログラミングしなくてよいため、ドライバやスタブのテストのための作業をしなくて済むという効果も奏し、一層プログラム作成を容易に且つ短時間で完成せしめることに寄与するところが大きい。

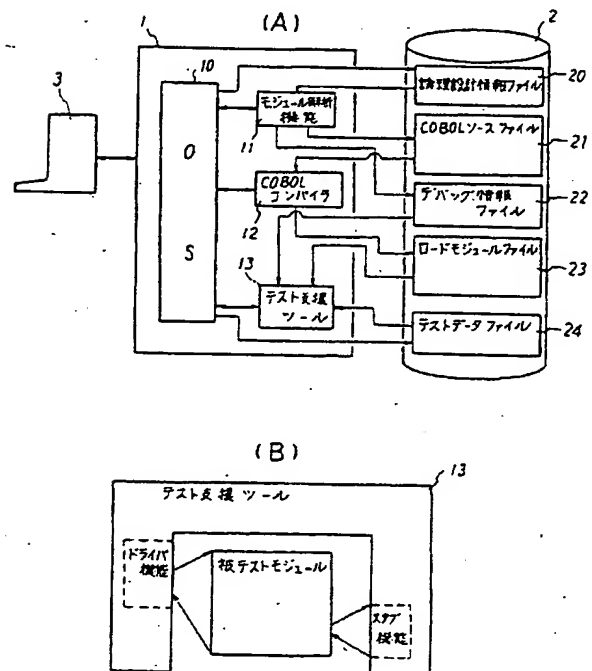
4. 図面の簡単な説明

オ1図は本発明の一実施例ブロック図、オ2図はオ1図実施例構成のテスト支援ツールの処理フロー図、オ3図はオ1図実施例構成に用いられる論理設計言語の説明図、オ4図はオ1図実施例構成のテスト支援ツールの動作説明図、オ5図は階層的モジュール構成プログラムの説明図、オ6図はオ5図構成のプログラムの開発手順説明図である。

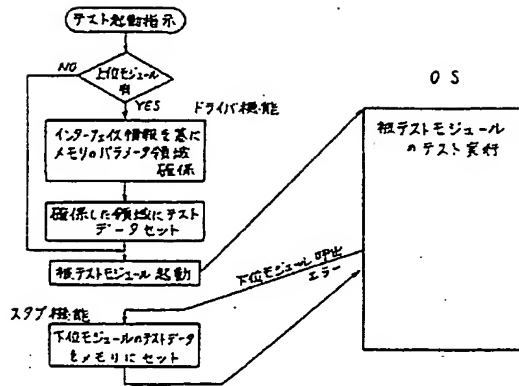
図中、1…コンピュータ(プロセッサ)、2…ファイルメモリ、3…コンソール、13…テスト支援ツール。

以上説明した様に、本発明によれば、複数のモジュールが階層的に構成されるプログラムのモジュール単体をプロセッサが実行してテストを行うプログラムの単体テスト方式において、該被テストモジュールを解析して該被テストモジュールに対する上位又は下位のモジュールの代替機能を行うテスト支援ツールを設け、該テスト支援ツールは該被テストモジュールが該上位モジュールから呼出される場合には、該被テストモジュールに渡すパラメータの領域をメモリに確保してテストデータをセット後該被テストモジュールを起動せしめ、該被テストモジュールが該下位モジュールを呼出す場合には、該被テストモジュールのテスト実行中に該下位モジュールが呼出されたことに応じて被テストモジュールに渡すテストデータを該メモリにセットして該被テストモジュールに渡すことを特徴としているので、関連する他のモジュールが完成していなくても被テストモジュールのテストを自動的に実行できるという効果を奏し、係るモジュールが階層的に構成されたプログラム

第1図



第 2 図



第 3 図

(A)

共通フェット足数

| 項目 | 区分 | 項目 | 形式 | 桁 |
|----|----|-------|-------|---|
| 10 | | 口座番号 | 9 (4) | |
| 20 | | 科目コード | 9 (4) | |
| 30 | | 氏名 | N (6) | |

(C)

モジュール定義 (モジュールB)

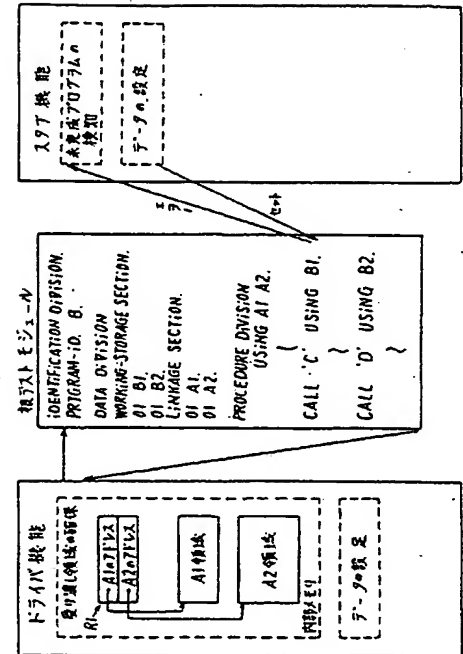
| 項目 | 区分 | 処理 | 処理条件 |
|----|----|-------------------|------|
| 10 | | | |
| 20 | | | |
| 30 | | 日誌 %DATE% C | |

(B)

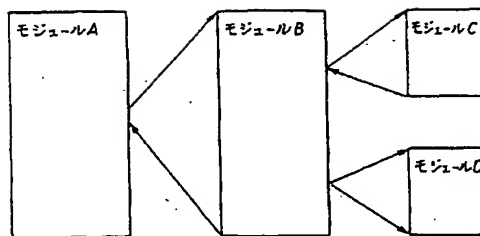
インターフェイス定義 (モジュールB)

| 項目 | 区分 | インターフェイス | 入力項目 |
|----|----|----------|-----------------------------|
| 1 | 5 | 管理管理 | GRP:G1 = A1 B G2 = A2 |

第 4 図



第 5 図



第 6 図

